

INDIVIDUALIZED SPEECH GENERATION UTILIZING
THE INTEL 8086 MICROPROCESSOR AND THE VOTRAX SC-01 VOICE SYNTHESIZER

A RESEARCH PAPER
SUBMITTED TO THE GRADUATE EDUCATIONAL POLICIES COUNCIL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

MASTER OF ARTS

by

TIMOTHY STUART McLAREN

ADVISER - PAUL R. ERRINGTON, PH.D.

BALL STATE UNIVERSITY

MUNCIE, INDIANA

MAY, 1983

Thesis
LD
2489
.Z9
1983
.M36

ACKNOWLEDGEMENTS

I would like to express my gratitude to three of my co-workers for their assistance: Mr. Tom Pettit for his technical advice about specifics of the programming, Mr. Steve Garnier for his aid with the use of the development system, and Mr. Bob Fuhrmann for the use of some of his tools and his advice regarding the construction of the hardware.

I am extremely grateful to my adviser, Dr. Paul Errington, for his cooperation and assistance in this entire project. We faced some very difficult circumstances caused by my living 1000 miles away from the campus. I was very impressed by Dr. Errington's dedication and problem solving ability, without which the project might not have been completed.

I would also like to thank my wife, Penny, for her assistance with the library research and for the final editing of the report, and of course, for enduring all that goes with having a spouse engaged in this activity.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	i
TABLE OF CONTENTS	ii
 CHAPTER	
I. PROBLEM	1
II. REVIEW OF RELATED LITERATURE	3
III. METHOD	6
Circuit Description	7
Program Description	10
Implementation	13
IV. FINDINGS	17
V. SUMMARY AND CONCLUSIONS	19
BIBLIOGRAPHY	21
APPENDIX A	23
Circuit Diagram	
APPENDIX B	27
Control Program Flow Chart	
APPENDIX C	31
Control Program Listing	
APPENDIX D	40
Test Vocabulary Listing	

CHAPTER I

PROBLEM

The object of the research conducted and presented herein was to develop a system consisting of both hardware and software around the Intel 8086 microprocessor and utilizing a Votrax SC-01 voice synthesizer microcircuit. The resulting system was to provide to the speech handicapped individual a synthesized voice which was capable of producing a readily intelligible individualized vocabulary. This synthesized voice was also to be provided at an economical cost in terms of power consumption for portability, ease of programmability of vocabulary, as well as at a low dollar cost for acquisition by the speech impaired individual.

In any society or group of people, a person who is different from the norm is at a disadvantage. The disadvantage is more serious if the difference is the inability to perform some function which is common to the norm.

In a society, the ability to communicate among the members is essential. In most intelligent societies the primary form of communication is verbal. The inability to speak thus places the individual at a great disadvantage. Even though there are other forms of communication such as written and visual (sign language) the inconvenience of writing and the limited knowledge by society in general of sign language is

indeed a handicap. The most common form of rapid, convenient, two-way communication, the telephone, is almost useless to many speech handicapped individuals.

The development of a portable synthetic voice for the speech impaired would allow the individual to communicate with people who do not know sign language; it would also enable the individual to utilize the telephone. The use of the telephone is especially important; it gives the handicapped person the independence of living alone while still having access to emergency services via the telephone. In order for the voice synthesizer to be an effective aid, it was important that it be portable, which requires small size, light weight, and low in power consumption to keep the battery requirements small. It was also desirable to have a device which could provide a personalized vocabulary (i.e. contain names of family and friends, addresses, technical terms for occupational use, etc.).

While there have been many applications for voice synthesizers (e.g., personal computers, video games, bank automatic tellers, etc.) there have been few significant developments relating to the portable, personalized, reasonably priced units. It is to this problem that this report is addressed.

The research herein consisted of the development, design and fabrication of hardware circuitry required to support an Intel 8086 microprocessor and a Votrax SC-01 voice synthesizer. The control programming had to be written, debugged and integrated with the hardware to complete the voice synthesizer system. This project did not pursue any other systems' development or the comparison of this system to any other.

CHAPTER II

REVIEW OF RELATED LITERATURE

Because of the specialized nature of the project which dealt with the Intel 8086 and the Votrax SC-01, the literature review was limited to technical periodicals. Also, because these devices have only been available for a few years, the review included only the most recent six years. The literature search concentrated on two principal areas, computer controlled voice synthesis and computer aids for the handicapped.

Previous research has fallen into one of three major categories. First, a few technical works have been written that deal with how computers synthesize speech. Such work has centered around vocal tract modeling and linear predictive coding. This type of information, while of educational benefit and interest to the investigator, had little bearing on this project since all of the functions discussed take place completely within the Votrax SC-01 speech synthesizer integrated circuit. The work of Gargagliano and Fons in particular was of interest.¹ Here was given a somewhat detailed description of the non-proprietary aspects of the internal operation of the Votrax SC-01.

¹Tim A. Gargagliano and Kathryn Fons, "Text Translator Builds Vocabulary for Speech Chips," Electronics, February 10, 1981, p. 118.

The second category of previous literature has dealt with descriptions of various aids for the handicapped. The majority of information available dealt with developments in the areas of mechanical aids. Very little work has been published that has examined aids for the verbally handicapped. Work that has been done has centered around either some form of manually controlled visual output or larger complex computer-based voice synthesis. Most of these descriptions were a few short paragraphs included in a general treatment of all types of aids for the handicapped. In an article by Andrew Thomas, the author did deal exclusively with aids for the vocally handicapped.² However, the emphasis here was on the device-to-individual interface and the difficulty of designing this interface for the handicapped. All of the devices discussed were of the visual output variety. This work should, however, be of benefit when designing various input devices for a speech synthesizer such as the one constructed for this project.

The final category of previous work has been in the publication of developments and announcements, often by corporations, which appear in the new products section of trade journals. The devices discussed in much of this work were preprogrammed with a standard vocabulary. One of the devices was a Votrax product (produced prior to the introduction of the SC-01 circuit) that could be user-programmed at the time of use by entering a phoneme code sequence, but this is hardly easy.³ All of the

²Andrew Thomas, "Communication Devices for the Nonvocal Disable," Computer, January 1981, p. 25.

³"Electronic Voice System Generates Messages for Vocally Handicapped," Electronics, November 10, 1977, p. 32.

devices described in this literature had prices in excess of \$1,000 with some over \$2,000. The investigator estimated that a mass produced (100 units or more) voice synthesizer based on the one constructed for this project would cost in the area of \$200 including some individual pre-programming for each unit.

The state of development of aids for the vocally handicapped was summed up as recently as 1981:

The stationary nature of current microcomputers tends to limit them to work station application. . . . The existing stationary systems cannot meaningfully address the conversational needs of individuals with severe speech impairments, but recently introduced portable and hand-held computers are opening up the potential for portable writing/notetaking and conversational communication aids. . . .

The major barrier to using microcomputers as communication aids, however is the custom interfacing needed to achieve optimum speed. . . ⁴

Since the system designed in this project uses only 12 switches for input, it should be easy to design various types of arrangements for individual handicaps not previously introduced in any other work in this area.

⁴Gregg C. Vanderheiden, "Practical Application of Microcomputers to Aid the Handicapped," Computer, January 1981, pp. 54-55.

CHAPTER III

METHOD

The Votrax SC-01 speech synthesizer integrated circuit was selected for this development project because of its phoneme based operation. English language speech has been divided into 46 basic sounds called phonemes. Words are formed by selecting phonemes in the proper sequence to produce the desired sound. This type of operation allows any word, including proper nouns, occupational peculiar terms, and selected phrases to be programmed using the phoneme codes. The basic operation of the system was to retrieve the stored phoneme codes and output them one at a time to form the selected word or phrase. The selection is made by numeric input on a keyboard. The vocabulary and the operating program were stored in two non-volatile memory integrated circuits (EPROM's), and the system was controlled by an Intel 8086 microprocessor.

The Intel 8086 microprocessor was selected as the central element for this project because of the availability of an Intel development system to support the checkout and debugging of the voice synthesizer. The development system allowed the integration of the program into the hardware without the need to program the EPROM's until both hardware and software were working as intended. The in-circuit emulation feature (ICE-86) permitted single step operation through trouble areas of the

program, execution of small sections of code for checking proper operation of the code segment and/or portions of the hardware circuitry, and a back trace of events prior to a termination (either intended or failure-created).

The Intel 8155 IO/RAM was chosen because of its familiarity to the investigator, and because it had sufficient capacity in both RAM and input/output ports to support the proposed system. The remainder of the components were determined from design requirements to support the selected devices.

Circuit Description⁵

The Intel 8284A (U1) provided the necessary timing and control signals for the 8086 microprocessor. A 15 mega-Hertz crystal was used to drive the clock circuit to produce a 5 mega-Hertz square wave for the system clock. An R-C network provided a reset signal through the 8284A as power was initially applied when turning on the synthesizer. The ready circuitry was not utilized in this design and was permanently wired in the ready state.

Because the 8086 multiplexes the address and data on the same lines, two 8282 (U6 and U7) eight bit address latch circuits were used to provide 16 lines of stable address inputs to the memories during memory access operations. The 8086 microprocessor utilizes 20 address lines in order to access one mega-byte of memory. Due to the limited size of memory in this system the four most significant address lines

⁵Refer to the circuit diagram in Appendix A in the following discussion.

were not used. The next four address lines (A/D 12 through A/D 15) were used for turning on various IC's through chip enable inputs. The address accessed upon system reset (FFFF0) required the use of an inverter in the chip enable not (\overline{CE}) line for the 2716 (U11 and U12) EPROM's to allow system activation. The 8086 address latch enable output (ALE) was used to latch the address from the address/data bus into the 8282 latch circuits.

The two 2716 EPROM's provided 4096 bytes of permanent storage for the control program and vocabulary. A proper latched address and a memory read not (\overline{RD}) output signal from the 8086 caused two bytes of data to be placed on the 16 line data bus. The least significant address bit (A/D 0) was not used for either 2716 since this bit only determines the high byte or low byte and the 8086 utilizes both at the same time and selects the desired byte internally.

The Intel 8155 (U10) IO/RAM circuit provided the interface between the processor, and the keyboard and the Votrax SC-01. Because the 8155 uses multiplexed address/data lines, similar to the 8086, it contains a set of internal latches for the address, controlled by the address latch enable signal. The IO/\overline{M} input selects whether the circuit input/output ports are accessed or the 256 byte random access memory is utilized. Because the 8086 control output is M/\overline{IO} , an inverter was used to provide the proper signal for the 8155. The read not (\overline{RD}) and write not (\overline{WR}) inputs are directly controlled by the 8086 and determine the direction of flow for the data for both the IO ports and RAM.

Port A on the 8155 was used as an output port to supply the phoneme codes to the Votrax SC-01. Port B was used for inputting the

return signal from the keyboard switches. Port C was used to output control signals to strobe the phoneme codes into the Votrax SC-01 and place select signals onto the keyboard select lines.

The keyboard consisted of 12 switches--8 on select line 0 and 4 on select line 1. The four switches on select line 1 were in parallel with the first four switches on select line 0 allowing the return lines to be limited to a total of eight, requiring only one input port. All eight return lines were individually tied to the positive five volt power supply through 1k ohm resistors. The select line signal was a zero voltage which dropped the appropriate return line to zero when a key was pressed. This approach provided a more sure and accurate response than allowing the return lines to float when not switched to a select line.

The Votrax SC-01 (U13) speech synthesizer circuit received the phoneme code inputs from the 8155 output port A. The two most significant bits of the phoneme code byte were used to vary the pitch of the output sound, but were not TTL logic level inputs. Thus the two transistor (Q1 and Q2) circuits had to be used to raise the TTL output from the 8155 to a level acceptable to the SC-01. The transistor circuitry inverted the 8155 output but the inversion was compensated by changing the two bits in the vocabulary coding. The MCX and MCRC inputs were used to control the speed and overall pitch of the sound output. The acknowledge not/ready ($\overline{A/R}$) output was used through an inverter to supply the test not (\overline{TEST}) input for the 8086. After the program output the control code (through the 8155 port C), a WAIT instruction was executed and the 8086 waited until the SC-01 indicated its readiness for the next

phoneme code by placing the appropriate signal on the test not ($\overline{\text{TEST}}$) pin. The sound output from the SC-01 was amplified by the LM 386 (U14) low voltage audio power amplifier and output to the speaker.

Because the 8086 is a 16 bit parallel processor, two 2111A-4 (U8 and U9) 256 X 4 bit RAM circuits had to be added in parallel to provide for stack storage of the most significant byte of address pushed into the stack when subroutines were called.

The two NAND gates and the two tri-state buffers were added during the system debugging and are discussed in section IV of this paper.

Program Description

The program had to perform five basic functions to produce the desired results:

1. Initialize certain quantities following the power on reset.
2. Scan the keyboard, convert the input switch signal to a binary number and temporarily store the inputted number.
3. Convert one, two or three digit stored binary coded decimal numbers to binary.
4. Find the desired word or phrase storage location.
5. Output the phoneme codes.

The program was written in 8086 assembly language to allow the required degree of control over the memory and input/output functions. The program listing containing the addresses and object code listings, and the pure object code for use by the actual system were assembled by the ASM 86 routine of the Intel development system. A copy of the assembled listing, including comments, is presented in Appendix C. The program flow chart is included as Appendix B.

The vocabulary consisted of a string of phoneme codes for each word. Each string was preceded by a byte containing the number of phonemes contained in a word. The phrases were constructed by listing the two byte address of each of the words used to make up the phrase. The first byte of each phrase contained twice the number of words in the phrase. A copy of the listing for the short test vocabulary used for each system check out is contained in Appendix D.

The program and vocabulary were assembled separately, and were combined during the execution of the LINK 86 routine on the development system. The origin of all of the program segments was placed at zero for assembly with the actual locations loaded during execution of the LOC 86 routine.

The program was entered by a jump to start instruction placed at the FFF0 reset address location by the RESET segment. Reset occurred when the power to the system was switched on. Once begun, the program continued to operate until power was turned off. The first lines of program (lines 17 through 41) were instructions for the assembler and were not executable code. Execution began by sending an I/O port control code to the command register of the 8155 circuit. A procedure (BEEP) was then called to output a beep sound to indicate that the system had been activated and was ready to receive input.

Keyboard Scan

The keyboard scan began initially, or after a CLEAR entry or an error, by clearing (filling with zeros) all of the temporary storage locations, digits entered storage, and the repeat indicator. This was

then followed by the keyboard scan instructions, which the program returned to after a successful key entry, or after completion of the output for a word or phrase. The SILENCE procedure was called to quiet the output from the Votrax SC-01 and the number of digits from the previous entry was returned from memory storage to register storage.

The keyboard was scanned by the two program portions called S0 and S1. These two segments alternately placed a signal on one of the two keyboard scan lines and then looked at the return lines to determine if an entry was made. If an entry was detected it was held in register storage and the DELAY procedure was called to provide for contact debounce. The scan was then repeated and the return lines checked against the previously detected entry. If the two entries matched, the entry was assumed valid and the program continued on to process the entry. Otherwise, the scan proceeded to the alternate scan line.

Convert Input to Binary, Check and Store

Upon receiving a valid input from scan line S1, the program checked for the entry of the ENTER key to begin execution of the preparation for output phases of the program, or the CLEAR key to output an error sound (BUZZ) and return to the initialization of the scan portion of the program. If the entry was any other key on scan line S1 or any key on scan line S0, the program utilized a rotate right loop to determine the numeric value of the key input from the position of the input bit. After the numeric value was obtained, the BEEP procedure was called to indicate a successful entry. The program then checked to see how many digits had previously been entered, and stored the current digit

in the appropriate storage byte. If three digits had already been entered and another digit entry was attempted, the program went to the error output of a buzz and then returned to reinitialize the scan portion. When the entered digit had been stored, the program incremented the input digit count and then called the DELAY procedure. After the delay, the keyboard was checked to determine that the input key had been released before proceeding to scan the keyboard for the next entry.

Execution After ENTER Key Depressed

When the ENTER key was pressed the total number of digits entered was saved in memory for later use if the word or phrase was to be repeated. The digit count was then used for converting the binary coded decimal entry to binary. If the digit count was three then the first digit entered was multiplied by 100 by a series of left rotations and additions to the total register. The second digit was multiplied by 10 and added to the total and then the third digit was added to the total. Two and one digit entries were processed in a similar fashion.

The vocabulary was arranged with the phrases first, starting with zero, followed by the individual words, starting with 100. The program first checked to see if a number greater than the largest vocabulary number was entered. If so then a buzz sound was issued and control was returned to the initialization of the scan routine. Otherwise a check was made to determine if the entry was for a word or phrase. If the entry was for a single word, 100 was subtracted from the entry number (to skip the phrases) and the address of the first single word was loaded into a register (SI). The FINDSTART procedure was called and returned

the starting address for the desired word. The WORDOUT procedure was then called to output the word's phoneme codes to the Votrax SC-01. After completion of the word the program returned to scan the keyboard. If a phrase was selected the address of the first phrase was loaded into the SI register and FINDSTART was called. The first byte at the return address contained twice the number of words used by the phrase (two bytes were required for each word address). The word count was decremented as each word was output and when all words were completed, control returned to scan the keyboard. The remaining bytes of the phrase code contained the address for each word used by the phrase. This eliminated the need to search the vocabulary words for each word in the phrase. Each word was then output by the WORDOUT procedure.

Procedures

FINDSTART: This procedure received input of the base address for the word or phrase section and the number of the word or phrase from the bottom that was to be selected. The procedure retrieved the first byte at each successive word or phrase starting address and added the contents (quantity of bytes used for the word or phrase) plus one to the current address to obtain the starting address of the next word or phrase. The word or phrase number was decremented and the climb up the addresses continued until the selected word or phrase was reached. This address was then returned via the SI register.

WORDOUT: This procedure received the starting address of a word of vocabulary. It retrieved the first byte at the supplied address, which was the number of phonemes to be output, and decremented it as

each phoneme was output. When this quantity reached zero the procedure was complete. Each successive byte, which was a phoneme code, was then output via the 8155 port A to the Votrax SC-01. The WAIT instruction preceded the output of each phoneme code and held up output until the SC-01 indicated readiness for a new phoneme code. After each phoneme code was placed into port A, a pulse was sent to strobe the phoneme code into the SC-01 via bit 6 of port C of the 8155.

DELAY/PAUSE: These procedures consisted of a loop to decrement a number until zero was reached. DELAY started with a larger number and provided a time delay of approximately 10 milliseconds to slow the processing sufficiently for interface to the human operator. PAUSE provided a very short delay that was used to provide a longer duration strobe pulse for the SC-01.

BEEP/BUZZ/SILENCE: These procedures output a single phoneme code to the Votrax SC-01 via the 8155 port A, waited for the SC-01 ready indication and strobed the SC-01. BEEP and BUZZ were used to provide an audible output indication for some action, and SILENCE was used to shut off any audio output when no specific sounds were desired.

Implementation

The hardware circuitry was built on a bread-board using a combination of soldering and wire wrap. The wiring was checked with an ohmmeter and then the integrated circuits were installed in the wired sockets. The program and test vocabulary were entered into disc files using the Intel development system. The files were then assembled, linked and located by ASM 86, LINK 86, and LOC 86 respectively also on the development system.

An in-circuit emulator (ICE-86) was used to couple the bread-boarded hardware to the development system. The emulator allowed for check out of the hardware and debugging of the program prior to having programmed EPROM's for the circuit.

CHAPTER IV

FINDINGS

During the check out and debugging of the system it was discovered that access to the two output ports (A and C) on the 8155 was not possible. The cause was traced to an operating characteristic of the 16 bit 8086 processor. When odd addresses are specified the data for those addresses appears on the high order byte lines (A/D 8 through A/D 15). The addresses for ports A and C were 1 and 3 respectively and the 8155 was only connected to address/data lines 0 through 7. This meant that data for these two ports was not available to them. The easiest correction for this problem appeared to be to use an even address to access ports A and C. This required that an unused higher order address bit (A/D 8) be used and combined with bit A/D 0 only when output to ports A or C was required. This was accomplished by the addition of two NAND gates and an inverter to connect line A/D 8 to line A/D 0 when an input/output signal and an address latch enable signal were present. See Appendix A. The port addresses for ports A and C in the program had to be changed to accommodate the new circuitry and the even address requirement. The address for port A was changed from 8001 hexadecimal to 8100 hexadecimal, and for port C from 8003 hexadecimal to 8102 hexadecimal.

The preceding change resulted in the ability to output correctly through ports A and C but when an attempt was made to read from port B or the RAM located in the 8155 the information on bit A/D 0 was not accessible. This was caused because the previous addition of the NAND gate in the A/D 0 line prevented signal flow on this line from the 8155 to the data bus. In order to correct this problem two tri-state buffers were added to bypass the NAND gate during read operations. See Appendix A.

After the above corrections were made and a short between two socket pins had been found and eliminated, the system worked well and provided intelligible output of the test vocabulary. However, entry of the number zero or a repeat entry of the ENTER caused the program to malfunction. The program was corrected to allow for entry of zero, and modified to provide for a repeat of the previously entered word or phrase by another entry of the ENTER key. The repeat was cancelled by the entry of any other key.

The new and final program version (see Appendix C) was then checked out and found acceptable. Several new words were programmed into the vocabulary for familiarization of the investigator with the phoneme coding process. The entire hardware/software system was then considered to be successful.

CHAPTER V

SUMMARY AND CONCLUSIONS

During the course of this project a hardware circuit was designed and constructed to connect with and support the Intel 8086 microprocessor in controlling the Votrax SC-01 speech synthesizer. Although some problems were encountered during the check out phase of the project these were successfully overcome. The program, written in Intel 8086 assembly language, was debugged and worked as designed to provide the appropriate series of phoneme codes to produce an intelligible vocal output in response to a numerically selected input. The completed system was demonstrated to function as intended and proved that the Intel 8086 could be utilized to operate a Votrax SC-01 speech synthesizer to provide a substitute voice for the speech handicapped.

The construction and operation of the system, however, proved to be somewhat difficult because of the use of a 16 bit microprocessor to provide an 8 bit output to the SC-01. The best example of this is the extra circuitry and program modifications required to get the data on the lower byte when the odd port addresses were needed. The Intel 8086 also requires several support circuits, such as the clock generator and address latches, in order to have minimal function. Added to this is the extra RAM required to store the higher order byte of address during stack operations. These requirements added 5 integrated circuits

to the design to support the 16 bit system. Since only eight bits of data are required, and speed is of little consideration in this application, the Intel 8086 has much more capability than is needed and creates a much larger system than would otherwise be necessary.

The program written to operate this system is not as efficient as it could be, since no effort was made in that direction. The object of this project was to develop a working system to verify that the Votrax SC-01 could be operated by an Intel 8086 based dedicated micro-computer. An experienced programmer should be able to make many improvements in the program to increase the efficiency and especially reduce the lines of code required. Reduced code requirements would of course allow for an increase in the vocabulary capacity of the system.

While the system does work, it is very large in terms of the number of parts (power consumption) required and cumbersome in operation because of the 16 bit operation. The trend in microcircuit technology is towards more circuitry in a single package, and several multi-function circuits are becoming available. Some of these contain a microprocessor, RAM, I/O and some even have EPROM capability all in the same integrated circuit. The control program used in this project should be readily adaptable to some other type of hardware, since the functions would be essentially the same. Because of this and the lower number of components required, it should be fairly simple to adapt one of the new multi-function microprocessor circuits to this application utilizing only three or four integrated circuits.

BIBLIOGRAPHY

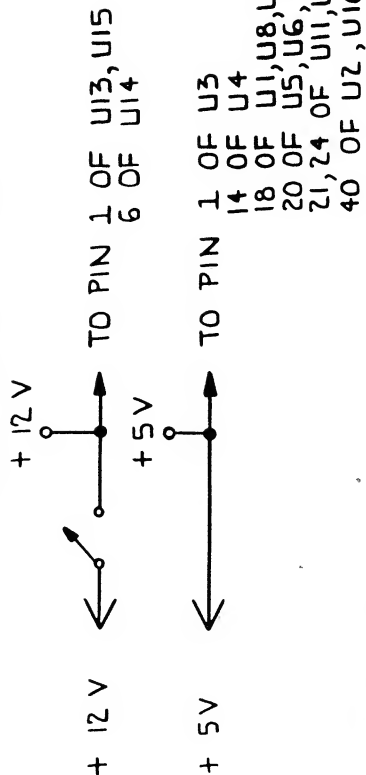
- Arnold, William F. "Signal Processor for Speech Synthesis is Programmable." Electronics, October 11, 1979, pp. 41-42.
- Bassak, Gil. "Phoneme-Based Speech Chip Needs Less Memory." Electronics, July 3, 1980, pp. 43-4.
- Burke, M. R., Komut, J., and Olive, J. P. "Speech Synthesis." Bell System Technical Journal, September 1981, pp. 1621-31.
- "Computerized Device Speaks for Handicapped Youngsters." Journal of the Acoustical Society of America, June 1976, pp. 1520-1.
- Crook, Sharon B. "Realistic Speech from Low-Cost Electronics." Machine Design, February 11, 1982, pp. 75-81.
- "Electronic Voice System Generates Messages for Vocally Handicapped." Electronics, November 10, 1977, pp. 32-33.
- Gargagliano, Tim A., and Fons, Kathryn. "Text Translator Builds Vocabulary for Speech Chip." Electronics, February 10, 1981, pp. 118-121.
- Hamilton, Pamela. "Communicators Help the Handicapped." Electronics, June 8, 1978, pp. 94.
- iAPX 86, 88 User's Manual. Santa Clara, California: Intel Corporation, August 1981.
- MCS-85 User's Manual. Santa Clara, California: Intel Corporation, 1977.
- Morse, Stephen P. The 8086/8088 Primer. 2nd Ed. Rochelle Park, New Jersey: Hayden Book Company, 1982.
- "New Device Talks for Vocally Handicapped." Instrumentation Technology, January 1978, p. 10.
- Newell, A. F. "Communication Aids for People with Impaired Speech and Hearing." Electronics and Power, October 1977, pp. 821-27.
- "One Chip Speech Synthesizers Cut External Parts Count to 2." Electronic Design, April 15, 1982, pp. 300.

- Pickvance, R. "Speech Synthesis: The New Frontiers." Electronic Engineering, July 1980, p. 37.
- Rector, Russell, and Alexy, George. The 8086 Book. Berkeley, California: Osborne/McGraw-Hill, 1980.
- SC-01 Speech Synthesis Data Sheet. Troy, Michigan: Votrax Corporation, 1980.
- Thomas, Andrew. "Communication Devices for the Nonvocal Disabled." Computer, January 1981, pp. 25-30.
- "Unlimited Vocabulary Fits on OEM Boards and Integrates into Computer Systems . . ." Electronics, June 5, 1980, pp. 42-43.
- Vanderheiden, Gregg C. "Practical Application of Microcomputers to Aid the Handicapped." Computer, January 1981, pp. 54-61.
- "Vocally Handicapped Get Two New Communicators." Machine Design, December 8, 1977, p. 39.
- Zimmerman, Mark D. "Technology for the Handicapped." Machine Design, April 8, 1982, pp. 38-43.

APPENDIX A

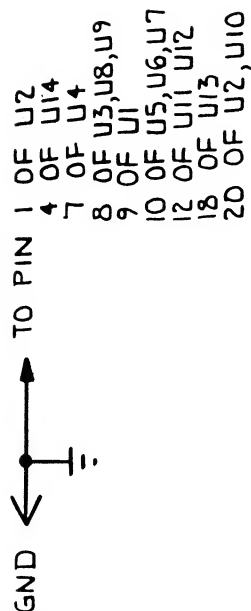
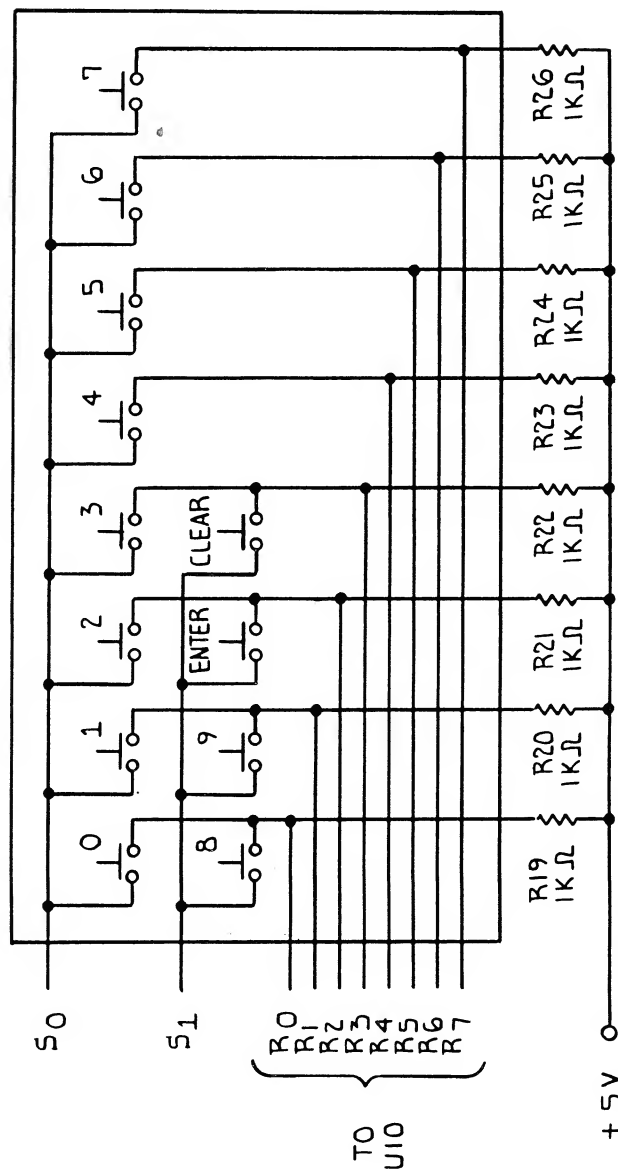
CIRCUIT DIAGRAM

POWER SUPPLY DETAIL



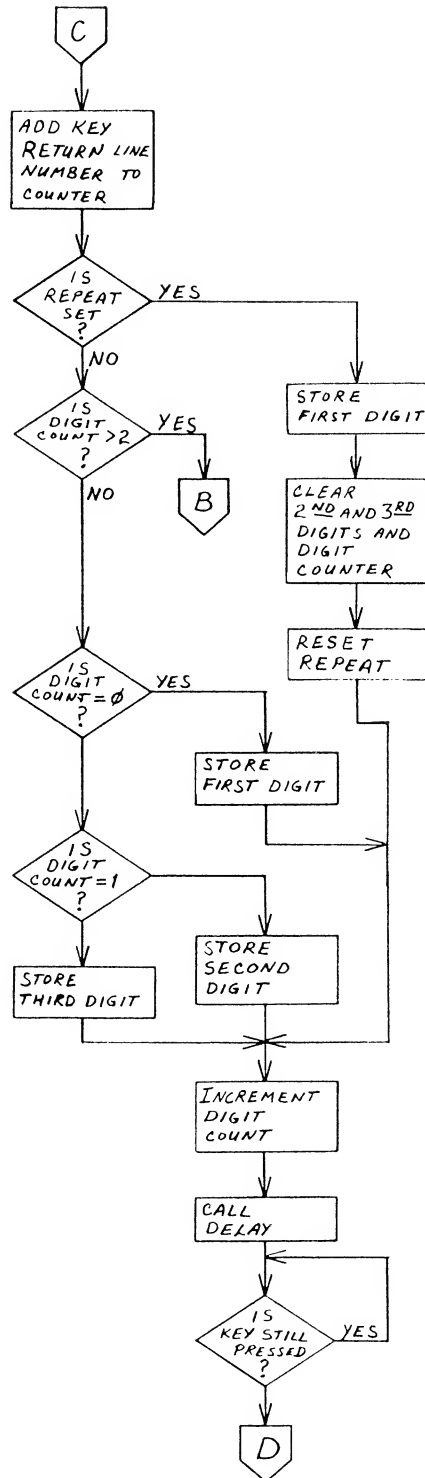
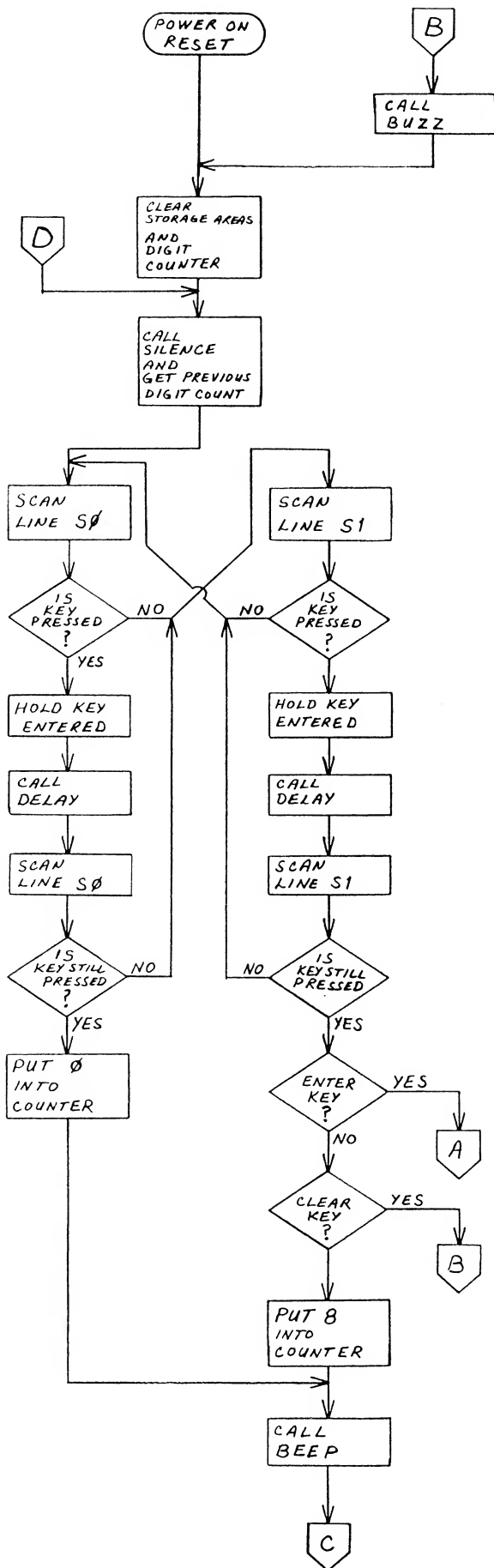
- NOTES —
1. POWER SUPPLY PIN (+ VOLTAGE) SHOULD HAVE 0.1UF CAPACITOR TO GROUND LOCATED AS CLOSE AS POSSIBLE TO EACH I.C.
- REPEATED TO SHOW DETAIL BREAKOUT OF A/D₈ AND A/D₀ AND A/D₈ ARE PART OF A/D₀ — A/D₁₅ ADDRESS DATA BUS.

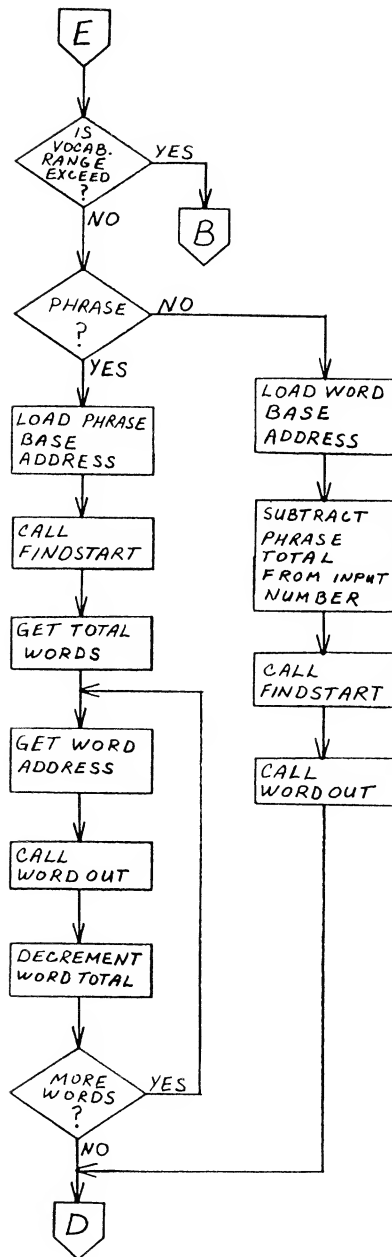
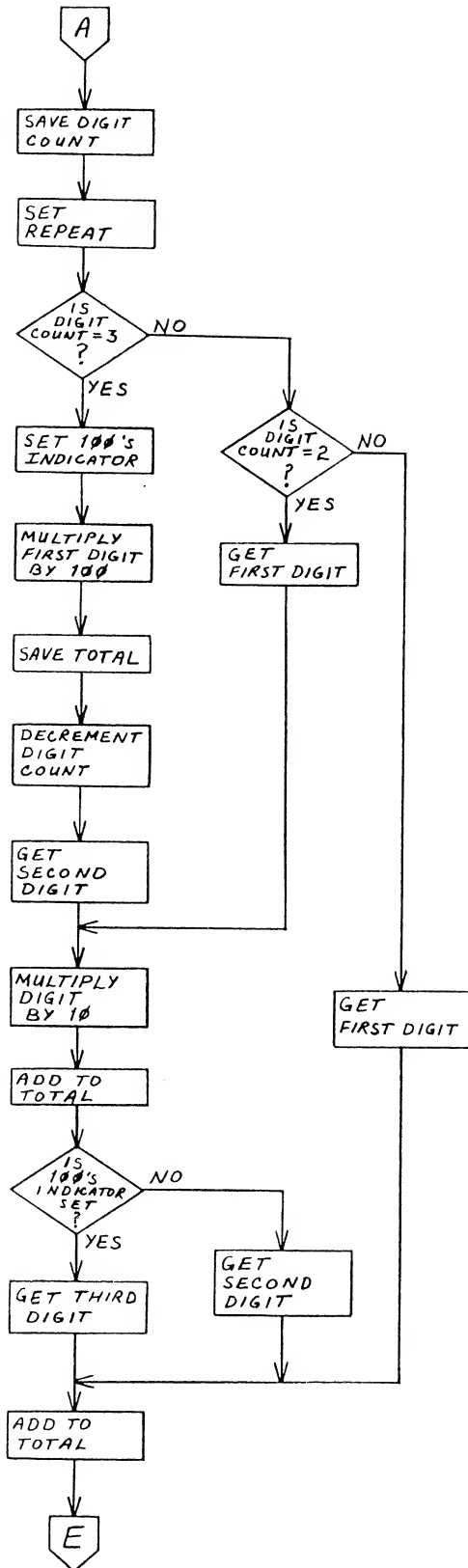
KEYBOARD DETAIL

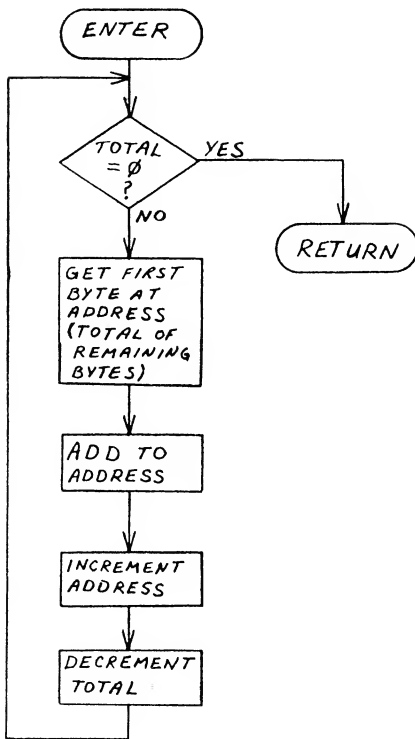
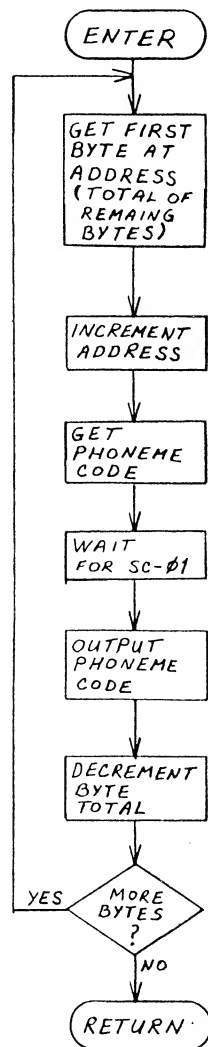
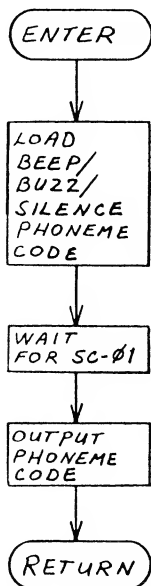
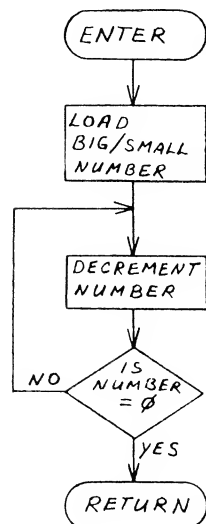


APPENDIX B

CONTROL PROGRAM FLOW CHART





PROCEDURESFINDSTARTWORDOUTBEEP/BUZZ/SILENCEDELAY/PAUSE

APPENDIX C

CONTROL PROGRAM LISTING

ISIS-II MCS-86 MACRO ASSEMBLER V2.1 ASSEMBLY OF MODULE SCCNT
OBJECT MODULE PLACED IN :F1:FINAL.OBJ
ASSEMBLER INVOKED BY: ASM86 :F1:FINAL.SRC

LOC	OBJ	LINE	SOURCE
		1	
		2	
		3	
		4	
		5	
		6	
		7	
		8	
		9	
		10	
		11	
		12	
		13	
		14	
		15	
		16	
		17	
		18	
		19	
0000	??	20	STOR
0000	??	21	SEGMENT
0002	??	22	ORG 0000H
0004	??	23	DB ?
0004	??	24	ORG 0002H
0006	??	25	DB ?
0006	??	26	ORG 0004H
0008	??	27	DB ?
0008	??	28	ORG 0006H
0010	??	29	DB ?
0010	??	30	ORG 0008H
		31	DB ?
		32	ORG 0010H
		33	DB ?
		34	ENDS
		35	PROG
		36	SEGMENT
		37	ORG 0000H
		38	ASSUME CS:PROG, DS:NOTHING
		39	ASSUME ES:STOR
		40	ASSUME SS:NOTHING
		41	LABEL FAR
		42	START
		43	MOV SP,0050H
		44	MOV AL,0DH
		45	MOV DX,8000H
		46	OUT LX,AL
		47	
		48	
		49	CALL RCEP
		50	

Master's research paper for RES 597

Tim McLaren - Candidate

Dr. Errington - Advisor

4-1 -83 0920 HRS.

8086/SC-01 Control Program.

The CS,DS,ES Registers will not be used and will be loaded with zero.

The low byte of stack and all temporary storage will be in the 8155 RAM. The high byte of the stack will be in the aux RAM.

INITIALIZE

NAME SCCNT

ORG 0000H

DB ?

ORG 0002H

DB ?

ORG 0004H

DB ?

ORG 0006H

DB ?

ORG 0008H

DB ?

ORG 0010H

DB ?

ENDS

PROG

SEGMENT

ORG 0000H

ASSUME CS:PROG, DS:NOTHING

ASSUME ES:STOR

ASSUME SS:NOTHING

LABEL FAR

MOV SP,0050H

MOV AL,0DH

MOV DX,8000H

OUT LX,AL

CALL RCEP

Indicate system is on

```

LDC OBJ          LINE      SOURCE
51      ;SCAN KEYBOARD
52      ;
53      ;
54      MOV AX,0H
55      MOV FSBYTE,AL
56      MOV SDBYTE,AL
57      MOV THRBYTE,AL
58      MOV BSTORE,AL
59      MOV RPT,AL
60      MOV BX,AX
61      ;and Disit counter
62      CALL SILENCE
63      MOV BL,BSTORE
64      MOV AL,0EH
65      MOV DX,8102H
66      OUT DX,AL
67      MOV DX,8002H
68      IN AL,DX
69      CMP AL,OFFH
70      JZ S1
71      MOV CL,AL
72      CALL DELAY
73      MOV AL,0EH
74      OUT DX,AL
75      MOV DX,8002H
76      IN AL,DX
77      CMP AL,CL
78      JZ CONVERT1
79      ;
80      MOV AL,0DH
81      MOV DX,8102H
82      OUT DX,AL
83      MOV DX,8002H
84      IN AL,DX
85      CMP AL,OFFH
86      JZ S0
87      MOV CL,AL
88      CALL DELAY
89      MOV AL,0DH
90      MOV DX,8102H
91      OUT DX,AL
92      MOV DX,8002H
93      IN AL,DX
94      CMP AL,CL
95      JZ CONVERT2
96      JMP S0
97      ;
98      ;
99      ;
100     ;CONVERT INPUT TO BINARY
101     CONVERT2:  NOT AL
102                CMP AL,04H
103                JZ HALF
104                CMP AL,03H
105                J7  CARRY

```

```

LOC OBJ          LINE  SOURCE
007B B108        106      MOV CL,08H ;The number is GT 7 so preload counter
007D 26C606100FD 107      MOV TEMPSTOR,OFDH ;Remember entry was line S1
0083 E80B90      108      JMP CONVERT1A ;Skip the next lines
                                ;
0086 F4D0        110      ;
0088 B100        111      NOT AL ;
008A 26C606100FE 112      MOV CL,0H ;Clear the counter reg.
0090 E85301      113      MOV TEMPSTOR,OFEH ;Remember entry was line S0
0093 E88201      114      ;
                                ;
0096 D0D8        115      CALL BEEP ;Indicate a successful key entry
0098 7228        116      CALL SILENCE ;Quiet the output again
009A FEC1        117      ;
                                ;
COUNT:          118      ;Convert the input line signal to
                                ;Binary by rotating the register to
                                ;the right and counting the number of
                                ;shifts until input signal is found.
121              119      RCR AL,1 ;Rotate the reg. by 1
122              120      JC STORE ;If a carry results go store the count,
123              121      INC CL ;else increment the count
124              122      JMP COUNT ;and go again
125              123
126              124
ERROR:            125      CALL BUZZ ;When needed make a buzz sound
127              126      JMP SCAN ;and return to SCAN
128              127
129              128
CLR:              129      MOV AX,0H ;
130              130      MOV FSTBYTE,CL ;
131              131      MOV SNIBYTE,AL ;
132              132      MOV THRD BYTE,AL ;
133              133      MOV BX,AX ;
134              134      MOV RPT,00H ;
135              135      JMP COMP ;
136              136
137              137
HALF:             138      JMP BEGIN ;
139              138
140              139
141              140
;STORE THE INPUT DIGIT
142              141      CMP RPT,01H ;
143              142      JZ CLR ;
144              143      CMP BL,2H ;Check for too many digits entered
145              144      JG CLR ;If yes go to ERROR to make a buzz
146              145      ;and restart SCAN
147              146
148              147      CMP BL,0H ;Is this the first digit entered?
149              148      JZ STORE1 ;If yes go to STORE1, else
150              149      CMP BL,1H ;Is this the second digit entered?
151              150      JZ STORE2 ;If yes go to STORE2, else
152              151      MOV THRD BYTE,CL ;Store it in the third digit space
153              152      INC BL ;Increment the digit counter
154              153      DELAY ;Delay for release of entry key
155              154
156              155
157              156
158              157
159              158
160              159
                                ;Check entry line to determine that the
                                ;entry key has been released (prevents
                                ;multiple entries)
KEYOFF:          160      MOV AL,TEMPSTOR ;Place entry line code into AL
                                ;
                                ;Output port address for keybd. scan lines
                                ;

```

LOC	OBJ	LINE	SOURCE
00EA EE		161	OUT DX,AL ;Output signal on scan line
00EB BA0280		162	MOV DX,8002H ;Input port address for keybd. return lines
00EE EC		163	IN AL,DX ;Get keyboard input
00EF 3CFF		164	CMF AL,OFFH ;If there is still an input from the keybd.
00F1 75F0		165	JNZ KEYOFF ;Keep looking until there isn't, else
00F3 E937FF		166	JMP SO ;Go scan for another entry
00F6 26880E0000		167	STORE1: MOV FSTBYTE,CL ;Store first digit input
00FB EBE1		168	COMP ;Go to COMP to complete digit storage
00FD 26880E0200		169	STORE2: MOV SNDBYTE,CL ;Store second digit input
0102 EBDA		170	COMP ;Go to COMP to complete digit storage
0104 EB98		171	ERC: JMP ERROR ;
		172	;
		173	;
		174	;
		175	;
		176	;
		177	;
		178	;
		179	BEGIN: MOV BSTORE,BL ;
0106 26881E0800		180	RPT,01H ;
010B 26C606060001		181	MOV AX,0H ;Clear AX and
0111 B80000		182	MOV DX,AX ;DX registers
0114 8BD0		183	MOV CX,AX ;and CX.
0116 8BC8		184	MOV BL,3H ;Have 3 digits been entered?
0118 80FB03		185	JNZ FSTTEN ;If not, skip *100
011B 7524		186	CL,1H ;Set Hundreds indicator
011D B101		187	MOV AL,FSTBYTE ;Get the first byte entered
011F 26A00000		188	AX,1 ;Multiply it by 100
0123 D1C0		189	ROL AX,1 ;
0125 D1C0		190	ROL AX,1 ;
0127 03D0		191	ADD DX,AX ;
0129 D1C0		192	ROL AX,1 ;
012B D1C0		193	ROL AX,1 ;
012D D1C0		194	ROL AX,1 ;
012F 03D0		195	ADD DX,AX ;
0131 D1C0		196	ROL AX,1 ;
0133 03D0		197	ADD DX,AX ;
0135 FEDB		198	DEC BL ;DX contains first digit entry *100 in binary
0137 B80000		199	AX,0H ;Decrement digit counter
013A 26A00200		200	MOV AX,0H ;Clear AX
013E EB0D90		201	AL,SNDBYTE ;Load the second digit entry into AL
		202	TIMESTEN ;Go multiply it by 10
0141 80FB02		203	CMF BL,2H ;Are there 2 digit remaining?
0144 7527		204	JNZ FSTONE ;If not skip to 1 digit process
		205	;
0146 B80000		206	AX,0H ;Clear AX
0149 26A00000		207	AL,FSTBYTE ;Get first digit entered
		208	;
014D D1C0		209	ROL AX,1 ;Multiply the number by 10
014F 03D0		210	ADD DX,AX ;
0151 D1C0		211	ROL AX,1 ;
0153 D1C0		212	ROL AX,1 ;
0155 03D0		213	ADD DX,AX ;DX contains digit *10 plus digit *100
		214	;
0157 B80000		215	MOV AX,0H ;Clear AX

SCCNT

MCS-86 MACRO ASSEMBLER

LOC	OBJ	LINE	SOURCE
015A	80F900	216	CL,0H
015D	7407	217	CMF
015F	26A00400	218	JZ
0163	EB0F90	219	MOV AL,THROBYTE
		220	JMP
		221	AL,SNDDBYTE
0166	26A00200	222	MOV
016A	EB0390	223	JMP
		224	AX,0H
016D	B80000	225	MOV
0170	26A00000	226	MOV AL,FSTBYTE
		227	ADD DX,AX
0174	03D0	228	
		229	
		230	OUTPUT A PHRASE
		231	
0176	83FA3F	232	DX,3FH
0179	7F89	233	ERC
		234	DX,00H
017B	EB2090	235	JMP
		236	WORD1 ;*****
017E	BE50B2	237	MOV
0181	E82800	238	CALL
0184	8BF0	239	MOV
0186	8A1D	240	MOV
		241	INC
0188	47	242	DI
0189	8A35	243	MOV
018B	47	244	INC
018C	8A15	245	MOV
018E	8BF2	246	MOV
0190	E82900	247	CALL
0193	F0CB	248	DEC
0195	80FB00	249	BL
0198	75EE	250	BL,0H
019A	E983FE	251	JNZ
		252	JMP
		253	SC
		254	
		255	OUTPUT A SINGLE WORD
		256	
019D	BE50B2	257	MOV
01A0	83EA00	258	SUB
01A3	E80600	259	CALL
01A6	E81300	260	CALL
01A9	E979FE	261	JMP
		262	SC
		263	
		264	
		265	
		266	*** PROCEDURES **
		267	
		268	FIND THE STARTING ADDRESS OF A WORD OR PHRASE
		269	
01AC		270	FINDSTRT PROC NEAR

LOC	OBJ	LINE	SOURCE
01ED 9B		326	
01EE EE		327	WAIT
01EF B020		328	OUT
01F1 BA0281		329	MOV DX,AL
01F4 EE		330	MOV AL,20H
01F5 B000		331	OUT DX,AL
01F7 E83700		332	MOV AL,00H
01FA EE		333	PAUSE
01FB 9B		334	CALL DX,AL
01FC 5A		335	OUT
01FD 58		336	WAIT
01FE C3		337	POP POP
		338	RET
		339	ENDP
		340	
		341	;BUZZ
01FF 50		342	
0200 52		343	PROC NEAR
0203 BA0081		344	AX
0206 9B		345	PUSH DX
0207 EE		346	MOV AL,12H
0208 B020		347	MOV DX,8100H
020A BA0281		348	MOV AL,20H
020D EE		349	OUT DX,AL
020E B000		350	MOV AL,00H
0210 E81E00		351	CALL PAUSE
0213 EE		352	OUT DX,AL
0214 9B		353	WAIT
0215 5A		354	POP DX
0216 58		355	POP AX
0217 C3		356	RET
		357	ENDP
		358	
		359	BUZZ
		360	
		361	
		362	;SILENCE THE SC-01 OUTPUT
		363	
		364	
0218 50		365	PROC NEAR
0219 52		366	AX
021A B03E		367	PUSH DX
021C BA0081		368	MOV AL,3EH
021F 9B		369	MOV DX,8100H
0220 EE		370	WAIT
0221 B020		371	OUT DX,AL
0223 BA0281		372	MOV AL,20H
0226 EE		373	MOV DX,8102H
0227 B000		374	OUT DX,AL
0229 E80500		375	MOV AL,00H
022C EE		376	CALL PAUSE
022D 9B		377	OUT DX,AL
022E 5A		378	WAIT
022F 58		379	POP DX
0231 53		380	POP AX

```

;Wait until SC-01 is ready
;Output phoneme code to SC-01
;Strobe the SC-01
;
;
;
;Wait until SC-01 is finished
;
;
;
;
;
;
;
;
;
;
;Put Z sound code into AL
;Load SC-01 output port address
;Wait until SC-01 is ready
;Output the phoneme code to the SC-01
;Strobe the SC-01
;
;
;
;Wait until the SC-01 is finished
;
;
;
;
;
;
;
;
;
;
;SILENCE THE SC-01 OUTPUT
;
;
;
;Put No Sound code into AL
;Load SC-01 output port address
;Wait until the SC-01 is ready
;Output the phoneme code to the SC-01
;Strobe the SC-01
;
;
;
;Wait until the SC-01 is finished
;
;
;
;

```

LOC	OBJ	SCNT	LINE	SOURCE	
0230	C3		381	RET	RET
			382	ENDP	ENDP
			383		
			384		! SHORT DELAY TO LENGTHEN STROBE
0231			385		
0231	50		386	PAUSE	PROC NEAR
0232	B88000		387		AX
0235	48		388		PUSH
0236	3D0000		389	D2:	MOV AX,0080H
0237	75FA		390		DEC AX
0238	58		391		CMP AX,0H
023C	C3		392		JNZ D2
			393		POP AX
			394	PAUSE	RET
			395		ENDP
			396		
			397		
			398	PROG	ENDS
			399		
			400		
			401	RESET	SEGMENT
0000			402		ORG 0000H
0000	EA0000----	R	403		JMP START
			404	RESET	ENDS
			405		END

ASSEMBLY COMPLETE, NO ERRORS FOUND

APPENDIX D

TEST VOCABULARY LISTING

ISIS-II MCS-86 MACRO ASSEMBLER V2.1 ASSEMBLY OF MODULE TVCNT
 OBJECT MODULE PLACED IN :F1:TSTVOC.OBJ
 ASSEMBLER INVOKED BY: ASM86 :F1:TSTVOC.ASM

LOC	OBJ	LINE	SOURCE
----		1	;Short vocabulary for test of
		2	;the 8086/88-01 control program
		3	;
		4	;3-29-83
		5	;Tim McLaren
		6	;
		7	NAME TVCNT
		8	;
		9	VOCAB SEGMENT
		10	;
		11	ORG 0220H
		12	;
		13	DB 06H,12H,21H,0BH,2BH,35H,37H ;ZERO
0220 06			
0221 12			
0222 21			
0223 0B			
0224 2B			
0225 35			
0226 37			
0227 04		14	DB 04H,2DH,32H,31H,0DH ;ONE
0228 2D			
0229 32			
022A 31			
022B DD			
022C 04		15	DB 04H,2AH,36H,37H,37H ;TWO
022D 2A			
022E 36			
022F 37			
0230 37			
0231 04		16	DB 04H,39H,2BH,3CH,29H ;THREE
0232 39			
0233 2B			
0234 3C			
0235 29			
0236 04		17	DB 04H,1DH,35H,34H,2BH ;FOUR
0237 1D			
0238 35			
0239 34			
023A 2B			
023B 05		18	DB 05H,1DH,15H,00H,29H,0FH ;FIVE
023C 1D			
023D 15			
023E 00			
023F 29			
0240 0F			
0241 06		19	DB 06H,1FH,0BH,09H,19H,03H,1FH ;SIX
0242 1F			
0243 0B			
0244 09			
0245 19			

LOC	OBJ	LINE	SOURCE
0246	03		
0247	1F	20	06H, 1FH, 02H, 00H, 0FH, 0AH, 0DH ; SEVEN
0248	06		
0249	1F		
024A	02		
024B	00		
024C	0F		
024D	0A		
024E	0D		
024F	04	21	04H, 05H, 05H, 29H, 2AH ; EIGHT
0250	05		
0251	05		
0252	29		
0253	2A		
0254	05	22	05H, 0DH, 15H, 00H, 29H, 0DH ; NINE
0255	0D		
0256	15		
0257	00		
0258	29		
0259	0D		
025A	04	23	04H, 2AH, 02H, 00H, 0DH ; TEN
025B	2A		
025C	02		
025D	00		
025E	0D		
025F	07	24	07H, 02H, 18H, 02H, 00H, 0FH, 0AH, 0DH, ELEVEN
0260	02		
0261	18		
0262	02		
0263	00		
0264	0F		
0265	0A		
0266	0D		
0267	07	25	07H, 2AH, 2DH, 02H, 00H, 23H, 18H, 0FH, TWELVE
0268	2A		
0269	2D		
026A	02		
026B	00		
026C	23		
026D	18		
026E	0F		
026F	07	26	07H, 39H, 3AH, 2AH, 2AH, 3CH, 29H, 0DH, THIRTEEN
0270	39		
0271	3A		
0272	2A		
0273	2A		
0274	3C		
0275	29		
0276	0D		
0277	04	27	04H, 2AH, 3CH, 29H, 0DH ; TEEN
0278	2A		
0279	3C		
027A	29		
027B	0D		
027C	04	28	04H, 22H, 00H, 02H, 1FH ; YES

LOC	OBJ	LINE	SOURCE
027D	22		
027E	00		
027F	02		
0280	1F		
0281	04	29	DB 04H, 0DH, 35H, 35H, 37H ; NO
0282	0D		
0283	35		
0284	35		
0285	37		
0286	02	30	DB 02H, 0BH, 2AH ; IT
0287	0B		
0288	2A		
0289	07		
028A	1E	31	DB 07H, 1EH, 15H, 2BH, 0DH, 0BH, 09H, 2AH; DARNIT
028B	15		
028C	2B		
028D	0D		
028E	0B		
028F	09		
0290	2A		
0291	03	32	DB 03H, 1BH, 15H, 2AH ; HOT
0292	1B		
0293	15		
0294	2A		
0295	05		
0296	2D		
0297	23	33	DB 05H, 2DH, 23H, 3AH, 19H, 1FH ; WORKS
0298	3A		
0299	19		
029A	1F		

		34	; VOCAB
		35	ENDS
		36	END

ASSEMBLY COMPLETE, NO ERRORS FOUND